

Real-Time 3D LiDAR-Based Obstacle Avoidance for UAV Using Sector-Filtering Algorithm

Krisma Asmoro^{*}, Silvirianti^{*}, Ryan Febriansyah^{**}, Soo Young Shin[°]

ABSTRACT

Unmanned aerial vehicle (UAV) gained a lot of attention because of its maneuverability in 3-dimensional (3D) and resulting in faster travel time to a certain destination. Therefore, UAV can be integrated to various human needs, especially in delivery service, search and rescue (SAR) mission, and utilized as military defense system. However, the fundamental problem of UAV is the need of proper navigation and obstacle avoidance to reach a destination safely. Diverse research of integrating 3D LiDAR for UAV navigation has been done to reduce complexity due to the high number of point cloud. This paper proposed a sector-filtering algorithm to improve existing real-time obstacle avoidance of UAVs using 3D LiDAR. The current proposed sector-filtering is proposed to take the advantages of 360° field of view (FoV) of 3D LiDAR, without sacrificing the computational resources. In addition, this paper implements 3D LiDAR avoidance with real-world experiment mounted on UAV. The results show that the obstacles have been successfully detected, and the avoidance maneuver is performed for the UAV in real time.

Key Words : 3D LiDAR, avoidance maneuver, obstacle detection, real-time, sector-filtering, Unmanned aerial vehicle(UAV)

I. Introduction

Thanks to their flexibility and high mobility, unmanned aerial vehicles (UAVs) have gained enormous popularity in recent years for real-world applications, i.e., military missions, search-and-rescue, and delivery^[1-5]. In line with the growth of its popularity, the autonomous navigation of UAVs has rapidly been under development and urging the system to be more applicable and efficient over time^[6-9]. Obstacle avoidance is one of the main challenges in UAV auton-

mous navigation. Therefore, providing the best possible solutions for efficient obstacle detection is needed since it is inseparable from obstacle avoidance.

Based on the type of sensors, generally, obstacle detection techniques can be divided into two categories: two-dimensional (2D) perception and three-dimensional (3D) perception. Some existing works proposed obstacle detection and avoidance techniques employing 2D-perception sensors^[10-12]. In [10], a real-time obstacle avoidance in dynamic environments where the mobile robot and the obstacle are moving

※ This work was partially supported by the IITP(Institute of Information & Communications Technology Planning & Evaluation)-ICAN(ICT Challenge and Advanced Network of HRD) grant funded by the Korea government(Ministry of Science and ICT)(IITP-2024-2022-00156394, 50%) and by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2024-RS-2024-00437190, 50%) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation)

♦ First Author : Kumoh National Institute of Engineering, Department of IT Convergence Engineering, krisma@kumoh.ac.kr, 학생회원

° Corresponding Author : Kumoh National Institute of Engineering, Department of IT Convergence Engineering, wdragon@kumoh.ac.kr, 종신회원

* École de Technologie Supérieure, silvirianti@gmail.com, 학생회원

** Research and development, Rovostech Co., Ltd., ryan@rovostech.com

논문번호 : 202404-058-E-RN, Received April 1, 2024; Revised July 9, 2024; Accepted September 4, 2024

was proposed by utilizing 2D LiDAR. In [11], a 2D LiDAR was employed to detect obstacles and translate online the essential geometric information of the obstacle-dense environments. In [12], a combination of mono-camera and 2D LiDAR was proposed for predicting the velocities and positions of surrounding obstacles through optical flow estimation, object detection, and sensor fusion. Although the techniques above that utilized 2D-perception sensors offered low computational complexity, which directly affects a short computational time, it has a significantly limited perspective that is only in 2D planes and thus results in one-sided field-of-view (FoV)^[13,14]. Unfortunately, the real environment is in 3D, which makes 2D-perception solutions insufficient, thus forcing solutions for obstacle detection from a 3D perspective to be provided.

Several works have proposed obstacle detection and avoidance employing depth cameras as 3D-perception sensors^[15-19]. In [15], a stereo frame-based camera was employed to avoid obstacles and safely navigate through an unknown cluttered environment. In [16], an active-sensing-based obstacle avoidance paradigm was proposed utilizing a stereo camera with an independent rotational degree of freedom to sense the obstacles actively. In [17], the ZED depth camera was utilized to detect and extract obstacles for robot navigation. In [18], a stereo camera system provided a point cloud of the environment to detect obstacles through time. In [19], a convex optimization framework was proposed utilizing a depth camera for obstacle detection that is vital for real-time autonomous vehicle operations. However, existing works that utilized depth cameras have limited detection range, limited FoV with less than 360 degrees, and environmental constraints, i.e., illumination, wind, temperature, atmospheric condition, environmental, and density. The limitations above notably contribute to lowering the detection accuracy. Dissimilar to depth camera characteristics, 3D LiDAR enables a broader range and FoV with its 360-degree view coverage^[25]. Moreover, the 3D LiDAR is keenly robust to environmental obstructions that improve detection accuracy due to the advantage of laser light compared to an image frame^[26]. By considering such conditions, em-

ploying 3D LiDAR can be one of the best solutions to improve the accuracy and FoV of obstacle avoidance in the real environment.

Some of the existing works employed 3D LiDAR for obstacle avoidance^[20-24]. In [20], a 3D LiDAR was employed to detect dynamic obstacles using an end-to-end sparse tensor-based deep neural network. In [21], a fully autonomous UAV system was proposed that enables flying safely in cluttered environments while avoiding dynamic obstacles using 3D LiDAR. In [22], a real point cloud from 3D LiDAR was augmented with synthetic obstacles for autonomous navigation. In [23], the utilization of 3D LiDAR for high-accuracy and high-efficiency 3D sensing was investigated and applied to intelligent transportation systems. In [24], an obstacle detection that addressed a partial scanning data availability issue was proposed employing 3D LiDAR. However, processing high-resolution and massive point cloud data from 3D LiDAR contributes to exponentially escalating computational burden. In consequence, computational time also rises exponentially. Considering the conditions above, those techniques can be challenging to apply in the UAV obstacle avoidance system where low computational time is required because of its high mobility^[27].

This study introduces a sector-filtering algorithm to reduce processing point clouds for obstacle detection using 3D LiDAR. To summarize the contribution of current study, Table 1 illustrated a comparison between existing 3D LiDAR literature. To be specified, the point clouds in three-dimensional space from 3D LiDAR are sectorized into eight zones. In this manner, all the point clouds in the 3D space can be mapped into the eight zones according to their location in degrees. Subsequently, an angular filter and

Table 1. Comparison with existing studies of 3D LiDAR in [20-24].

Reference	Avoid.	UAV	360°	SLAM	Training
[20]	×	×	✓	×	✓
[21]	✓	✓	×	×	×
[22]	×	×	✓	×	✓
[23]	×	×	✓	×	✓
[24]	×	×	×	×	✓
Proposed	✓	✓	✓	×	×

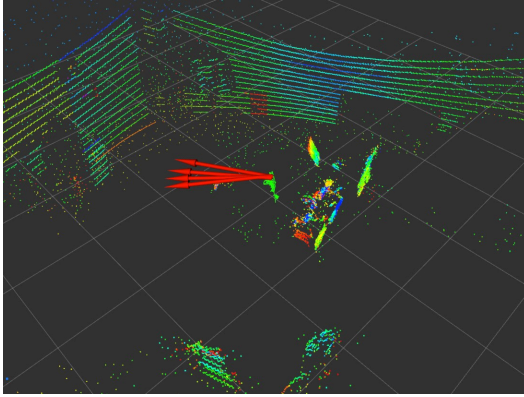


Fig. 1. Estimated LiDAR orientation (red arrow) and 3D point cloud (colored dots)

intensity filter are employed in each sector to reduce the dimension of point clouds from 3D to 2D points.

Thus, smaller data dimensions that affect lower computational complexity and computational time can be achieved.

1.1 Contributions

To the best of the authors' knowledge, few works employ 3D LiDAR and reduce its computational complexity for real-time UAV obstacle avoidance. Table 1 shows that current proposed system includes avoidance system with 360° degree LiDAR. In comparison with existing reference, the current approach does not require any training method in order to detect an object. Therefore, the main contributions are specified as follows:

- To highlight our contribution, this study considers a summary of existing algorithms with proximity-based UAV avoidance in Table 2. Based on the table, this study proposed sector filtering algorithm for 3D LiDAR because existing avoidance systems are limited to 2D point cloud. Moreover, 3D LiDAR serves a higher resolution and accuracy in terms of detecting object (up to 100 meters).
- Due to the large number of 3D point clouds compared to the 2D ones, this study proposed sector filtering algorithm^[28]. In comparison with related works of 3D LiDAR, the proposed algorithm could reduce computational time because of less

Table 2. Existing sensor available for Real-time obstacle avoidance [31].

Proximity Sensor	Range (m)	Res. (cm)	Acc. (cm)	PCL 2D	PCL 3D
TRanger Evo 600Hz	8	0.5	+/- 12	✓	×
TRanger Evo 60m	60	0.5-2	+/- 6	✓	×
RPLidar C1	12	0.15	+/- 0.5	✓	×
RPLidar A2M8	12	0.5	+/- 0.5	✓	×
RPLidar S1	40	3	+/- 5	✓	×
Proposed VLP16	100	H: 0.4 V: 2.0	+/- 3	✓	✓

number of point cloud. However, the tradeoff is that the trajectory will be longer than 3D-based SLAM because there is no mapping process. In exchange, UAV could react faster to avoid incoming obstacle.

- The proposed algorithm is integrated with 3D trajectory planning for UAVs.
- Performance of the proposed algorithm is validated through experiments in the real environment.

1.2 Paper Organization

The rest of this paper is organized as follows. The idea of the sector-filtering algorithm for real-time obstacle detection is introduced and described in Section II. Section III presents obstacle detection and avoidance using 3D LiDAR for UAVs. The proposed algorithm's experimental results and performance analysis are discussed in Section IV. Finally, this paper is concluded in Section V.

1.3 Notations

Matrices and vectors are denoted by upper-case and lower-case bold letters, respectively. The notations $(\mathbf{B})^T$ and $(\mathbf{B})^{-1}$ denote transpose and matrix inversion of $\mathbf{B}$, respectively. The notation $\mathbb{R}$ denotes a real number.

II. Sector-filtering Algorithm for Real-time Obstacle Detection

This section presents the proposed sector-filtering algorithm for real-time obstacle detection. The objective of the algorithm is to reduce the computational complexity of processing massive and high-resolution point cloud data from 3D LiDAR. The algorithm consists of data acquisition, point cloud conversion, point cloud filtering, and obstacle mapping.

2.1 Data Acquisition

The initial phase of the proposed scheme involves collecting detailed point cloud information of the environment employing the Velodyne VLP-16 3D LiDAR sensor, which is affixed to the UAV. This sensor scans the surroundings in a full 360-degree range, capturing a multitude of points every second. The amassed point cloud offers an intricate depiction of the UAV's surroundings, encompassing potential obstacles.

Without loss of generality, this study consider a similar approach with [25], where each LiDAR points is repre-sented as 3-Dimensional Cartesian coordinates (x, y, z) . Then, the overall coordinates of point k is expressed as follows

$$\begin{aligned} x &= \kappa \cos \theta \sin \phi, \\ y &= \kappa \cos \theta \cos \phi, \\ z &= \kappa \sin \theta, \end{aligned} \quad (1)$$

where κ , θ , and ϕ denote distance of a point from the origin, elevation angle, and azimuth angle of the 3D LiDAR, respectively.

Let us consider an object detected, where every part of the detected object is marked as a point cloud. Moreover, all the point clouds of the detected object can be denoted as $\mathbf{P} \in \mathbb{R}^3$. Assumed K point clouds formed as detected objects, a single point cloud $P_k \in \mathbf{P}$, where $k = \{1, \dots, K\}$, is located at x_k, y_k, z_k in a 3D environment $\mathbf{W} \in \mathbb{R}^3$. Due to the limitation of LiDAR based detection to laser misalignment, some of the laser points did not bounce back from environment, resulting in zero value of k -th point cloud^[24]. Let P_k be the variable to store a cartesian value whether an object is successfully detected or not^[29]. A point cloud is considered to be filled as a representative of the detected object when the $P_k = 1, \forall k \in K$ and considered to be empty when the $P_k = 0, \forall k \in K$, which can be expressed as

$$P_k = \begin{cases} x_k, y_k, z_k; & \text{if } x_k, y_k, z_k \notin \emptyset, \\ 0; & \text{otherwise.} \end{cases} \quad (2)$$

The generated point clouds $\mathbf{P}$ from the detected object in the environment $\mathbf{W}$ at time t can be expressed as

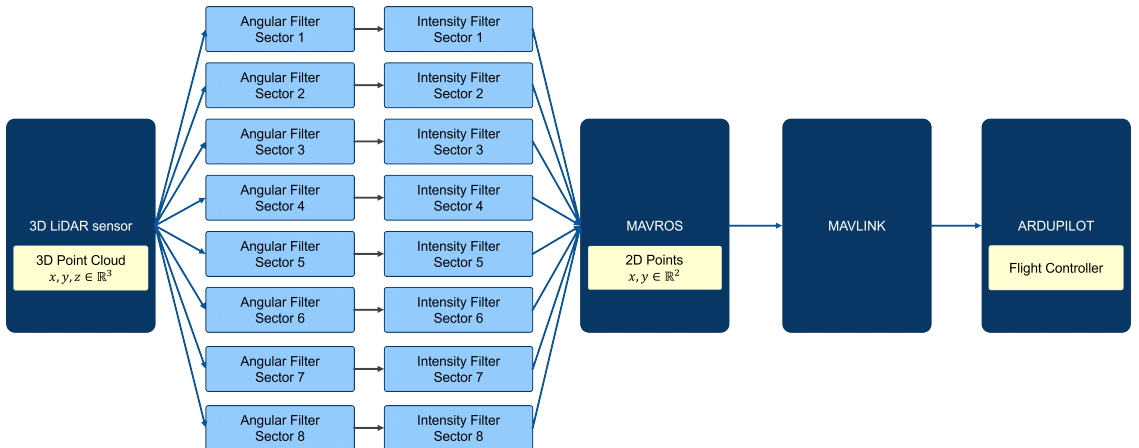


Fig. 2. Diagram flow of 3D LiDAR avoidance.

$$\mathbf{P} = \begin{bmatrix} x_1, y_1, \mathbf{z} & x_2, y_1, \mathbf{z} & \dots & x_K, y_1, \mathbf{z} \\ x_1, y_2, \mathbf{z} & x_2, y_2, \mathbf{z} & \dots & x_K, y_2, \mathbf{z} \\ \vdots & \vdots & \ddots & \vdots \\ x_1, y_K, \mathbf{z} & x_2, y_K, \mathbf{z} & \dots & x_K, y_K, \mathbf{z} \end{bmatrix}, \quad (3)$$

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_K \end{bmatrix}, \quad (4)$$

where x_K , y_K , and z_K denote the length, width, and height of the detected object, respectively. The data format of each returned LiDAR point is a 4-tuple formed by its coordinate with respect to the LiDAR coordinate frame $\mathbf{P}(x, y, z)$ as well as its intensity denoted by ρ .

2.2 Point Cloud Conversion

The raw point cloud data, denoted by $\mathbf{P}(x, y, z)$, collected from the 3D LiDAR sensor, as mentioned in the previous subsection, is in a binary file format. Unfortunately, data formed in binary format is not compatible with the Ardupilot robot operating system (ROS) message. Since this study employs ROS as a bridge for software control systems and hardware actuators, directly utilizing raw 3D point cloud data is unfeasible. Consequently, the raw point cloud data is required to be converted to a format compatible with the ROS message. In practice, the Point Cloud Library (PCL) software is employed in the conversion process to convert the raw Point Cloud data that is in binary format into a Point Cloud Data ROS format^[30]. At each time step, the raw point cloud data $\mathbf{P}(x, y, z)$ are published in the format of PCL ROS message as *pcl : : Point XYZ*^[29]

2.3 Obstacles Mapping

After the point cloud data have been successfully converted, all the detected obstacles are mapped and unified to acknowledge the location of the detected obstacles in the environment $\mathbf{W}$. First, all the detected obstacles at time t are mapped and gathered in the environment, denoted by $\mathbf{C}$. Assume the 3D LiDAR has successfully detected N obstacles; the obstacles map then can be expressed as

$$\mathbf{C} = \{\mathbf{P}_1, \mathbf{P}_2, \dots, \mathbf{P}_N\}, \quad (5)$$

where $\mathbf{P}_n$, $n = 1, 2, \dots, N$ denotes the generated point clouds $\mathbf{P}$ of each detected obstacle at time t . The generated point clouds of a single detected obstacle $\mathbf{P}_n$ are formed by K point clouds, which can be expressed as

$$\mathbf{P}_n = \{P_1, P_2, \dots, P_K\}, \quad (6)$$

where P_k , $k = 1, 2, \dots, K$ denotes the k -th point cloud of the single detected obstacle. All the generated point clouds can then be represented as unified point clouds from the detected obstacles. It is important to note that each generated point clouds $\mathbf{P}_n$ is not overlapped with other point clouds $\mathbf{P}_{j \neq n}$ which can be expressed as

$$\begin{aligned} \mathbf{P}_1 \cup \mathbf{P}_2 \cup \dots \cup \mathbf{P}_N, \\ \mathbf{P}_n \cap \mathbf{P}_{j \neq n} = \emptyset. \end{aligned} \quad (7)$$

Finally, all detected obstacles in the environment at time t can be expressed

$$\mathbf{C} = \bigcup_n^N \mathbf{P}_n, \quad (8)$$

where N denotes the number of detected obstacles at time t .

2.4 Point Cloud Filtering

In order to improve the detection accuracy, noise, and unwanted point clouds that are irrelevant to the detected obstacles must be removed through the point cloud filtering process. The filtration procedure includes utilizing statistical filters like outlier elimination and voxel grid down-sampling. These filters aid in eliminating undesired data points and diminishing the overall size of the point cloud data, simplifying the processing task. The filtering process generally consists of an angular filter and an intensity filter. However, a pre-processing filter is introduced to sharply reduce the noise in horizontal and vertical planes of the point clouds $\mathbf{P}$. The pre-processing for the horizontal plane is done by setting a threshold

based on the distance from each point to the origin, which can be expressed as follows

$$(x^2 + y^2 + z^2)^{1/2} \leq \delta_h, \quad (9)$$

where δ_h denotes the threshold distance in horizontal. Moreover, the pre-processing for the vertical plane can be done by setting a threshold based on the vertical location of each point, which can be expressed as follows

$$|z| \leq \delta_v, \quad (10)$$

where δ_v denotes the maximum vertical distance between the origin and each point. In addition, the noises and data outliers can be removed by employing a median filter, which can be expressed as follows

$$\hat{z}(x_j, z_j) = \mathbf{Med}_{(x_i, y_i) \in \Gamma} \{z(x_i, y_i)\}, \quad (11)$$

where $\hat{z}(x_j, z_j)$ denotes new vertical coordinate of the point at (x_j, y_j) , the $\mathbf{Med}\{\cdot\}$ operator is employed to calculate the median of a set of data, $z(x_i, y_i)$ represents the vertical point at coordinate (x_j, y_j) coordinate, and Γ denotes the neighbourhood centered on the point at coordinate (x_j, y_j) .

Angular Filter: The angular filter is designed to remove the noises and data outliers concerning the elevation angle (θ) and azimuth angle (ϕ) of the 3D LiDAR. Specifically, the minimum elevation angle is filtered by setting a threshold on the azimuth angle, which should be in the range of minimum and maximum azimuth angle, which can be expressed as follows

$$\theta_{\min} = \begin{cases} \theta_{\min} + \mu, & \text{if } \Phi_{\min} \geq \phi \geq \Phi_{\max} \\ \theta_{\min}, & \text{otherwise,} \end{cases} \quad (12)$$

where $\theta_{\min}$, $\Phi_{\min}$, $\Phi_{\max}$ and μ denote the minimum elevation angle, minimum azimuth angle, maximum azimuth angle, and constant parameter, respectively, similarly, the maximum elevation angle is filtered by set-

ting a threshold on the azimuth angle, which should be in the range of minimum and maximum azimuth angle, which can be expressed as follows

$$\theta_{\max} = \begin{cases} \theta_{\max} + \mu, & \text{if } \Phi_{\min} \geq \phi \geq \Phi_{\max} \\ \theta_{\max}, & \text{otherwise,} \end{cases} \quad (13)$$

where $\theta_{\max}$ denotes maximum elevation angle and μ is the constant parameter.

Intensity Filter: The intensity filter is designed to remove the noises and data outliers based on distance from each point to the intensity of the point clouds that should be in the range of minimum and maximum intensity, which can be expressed as follows

$$(x^2 + y^2 + z^2)^{1/2} = \begin{cases} \delta_h + \mu, & \text{if } \Lambda_{\min} \geq \Psi \geq \Lambda_{\max} \\ (x^2 + y^2 + z^2)^{1/2}, & \text{otherwise,} \end{cases} \quad (14)$$

where the term $(x^2 + y^2 + z^2)^{1/2}$ refers to distance from each point, δ_h is the threshold distance in horizontal, μ is the constant parameter, Ψ denotes the intensity of point clouds $\mathbf{P}$, $\Lambda_{\min}$ is the minimum intensity threshold for point clouds, and $\Lambda_{\max}$ is the maximum intensity threshold for point clouds. Furthermore, the intensity of point clouds in the vertical location can be filtered by setting a threshold with minimum and maximum intensity, which can be expressed as follows

$$|z| = \begin{cases} \delta_v + \mu, & \text{if } \Lambda_{\min} \geq \Psi \geq \Lambda_{\max} \\ |z|, & \text{otherwise,} \end{cases} \quad (15)$$

where $|z|$ refers to the vertical location of each point, δ_v denotes the vertical distance threshold, and μ is the constant parameter.

2.5 Remapping based on Point Cloud Filtering

The filtered point cloud data is subsequently uti-

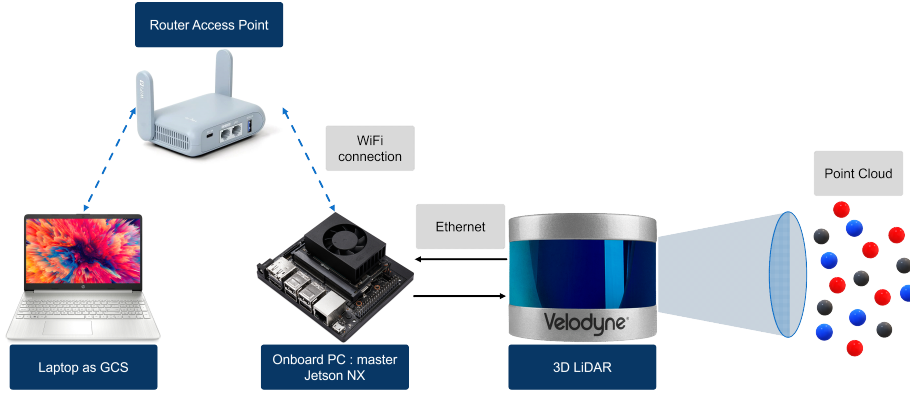


Fig. 3. Hardware configuration of the proposed 3D LiDAR UAV avoidance.

lized as inputs of a sectoring process, each delineating a distinct section of the environment. The sector is generated by applying a threshold to the point cloud data, where points falling within the threshold are assigned a value of 1, while those outside it are assigned a value of 0. The classified obstacle maps are combined to create a single map representing the entire environment.

2.6 Sectorization of Obstacles

Following the filtering stage, all the detected obstacles $\mathbf{C}$ are sectorized in order to classify their locations in the environment $\mathbf{W}$ from the FoV of UAV at time t . As previously mentioned in Section I, the leverage of employing 3D LiDAR is the 360° FoV that enables wider detection for the UAV. By utilizing the benefits mentioned earlier of the 3D LiDAR, the information about the detected obstacles can be more specified. In this study, the FoV of 3D LiDAR is sectorized into 8 sectors, which implies Sector 1 can be determined as $\mathcal{S}_1 \in [0^\circ, 45^\circ]$, Sector 2 as $\mathcal{S}_2 \in [46^\circ, 90^\circ]$, Sector 3 as $\mathcal{S}_3 \in [91^\circ, 135^\circ]$, Sector 4 as $\mathcal{S}_4 \in [136^\circ, 180^\circ]$, Sector 5 as $\mathcal{S}_5 \in [181^\circ, 225^\circ]$, Sector 6 as $\mathcal{S}_6 \in [226^\circ, 270^\circ]$, Sector 7 as $\mathcal{S}_7 \in [271^\circ, 315^\circ]$, and Sector 8 as $\mathcal{S}_8 \in [316^\circ, 360^\circ]$.

A single obstacle $\mathbf{P}_n(x, y, z)$ can be categorized as located in Sector 1,2,3,4,5,6,7, or 8 if its azimuth angle (ϕ), from the spherical coordinates (κ, θ, ϕ) which can be obtained by converting its cartesian coordinates (x, y, z), that is the angle in the x-y plane with respect to the positive direction of the x-axis, is in the range of $[0^\circ, 45^\circ]$, $[46^\circ, 90^\circ]$, $[91^\circ, 135^\circ]$, $[136^\circ, 180^\circ]$,

$[181^\circ, 225^\circ]$, $[226^\circ, 270^\circ]$, $[271^\circ, 315^\circ]$, or $[316^\circ, 360^\circ]$, respectively. Moreover, the location of obstacles can be detected, whether in the upper or below of the UAV, based on the elevation angle (θ), which is the angle to the positive direction of the z-axis.

III. Real-time Obstacle Avoidance and Navigation for UAV using 3D LiDAR

Following the obstacle detection employing the proposed sector-filtering algorithm with the 3D LiDAR, the UAV executes an avoidance maneuver in real time and continues navigating toward a target location. In order for the UAV to successfully navigate to the target location while avoiding obstacles along the way, the UAV navigation is divided into an obstacle avoidance stage and a navigation stage, which can be described as follows

3.1 UAV Real-time Obstacle Avoidance

Let us consider a scenario where a UAV equipped with a 3D LiDAR flies to a target location and meets obstacles along the way. The UAV detects the obstacles employing the proposed sector-filtering algorithm with the 3D LiDAR mentioned in Section II and avoids the detected obstacles in real time by employing a Kalman filter algorithm. The Kalman filter is employed because of its well-known ability to predict position accurately by minimizing error measurement from the noises, thus increasing navigation accuracy. In the Kalman filter algorithm, there are two phases which can be referred to as a prediction phase

and an update phase.

Prediction Phase: In the prediction phase, the Kalman filter assists the UAV in predicting its next position in the environment while acknowledging the surrounding obstacles at time t . In other words, the prediction phase is to estimate the avoidance maneuver for the UAV at each time t . The next position of UAV at time t can be predicted by

$$\bar{s}^{(t)} = s^{(t-1)} \mathbf{F}_{\text{UAV}} + v_{\text{UAV}}^{(t)} \mathbf{C}_{\text{ctr}}, \quad (16)$$

where $\bar{s}^{(t)}$ denotes a predicted state at t -th time, $s^{(t-1)}$ denotes the estimated state at $t-1$ -th time, $\mathbf{F}_{\text{UAV}}$ denotes state transition matrix representing motion model of the UAV, $v_{\text{UAV}}^{(t)}$ denotes velocity of the UAV at t -th time, and $\mathbf{C}_{\text{ctr}}$ denotes control-input matrix i.e., velocity or acceleration command input. Moreover, the prediction of state covariance matrix at t -th time, denoted by $\bar{\mathbf{P}}_s^{(t)}$, can be expressed as

$$\bar{\mathbf{P}}_s^{(t)} = \mathbf{P}_s^{(t-1)} \mathbf{F}_{\text{UAV}} \mathbf{F}_{\text{UAV}}^T + \mathbf{W}_{\text{noise}}, \quad (17)$$

where $\mathbf{P}_s^{(t-1)}$ is the predicted state covariance matrix at $t-1$ -th time and $\mathbf{W}_{\text{noise}}$ denotes the process noise covariance matrix.

Update Phase: Following the prediction phase, the update phase is designated to update the position of the UAV at each time t after the avoidance maneuver. The position of the UAV at time t can be updated by

$$s^{(t)} = \bar{s}^{(t)} + K^{(t)} \bar{\Lambda}^{(t)}, \quad (18)$$

where $s^{(t)}$ denotes estimated update state at t -th time. $\bar{\Lambda}^{(t)}$ is the estimation of avoidance variable at t -th time, which can be expressed as

$$\bar{\Lambda}^{(t)} = \mathbf{Q}^{(t)} - \mathbf{M} \bar{s}^{(t)}, \quad (19)$$

where $\mathbf{Q}^{(t)}$ represents the 3D LiDAR measurement at t -th time and $\mathbf{M}$ denotes the predicted measurement matrix of the generated 3D map from 3D LiDAR. $K^{(t)}$

represents the Kalman gain at t -th time, which can be formulated as

$$K^{(t)} = \bar{\mathbf{P}}_s^{(t)} \mathbf{M}^T \mathbf{L}^{(t)-1}, \quad (20)$$

where $\mathbf{L}^{(t)}$ denotes residual covariance, which can be expressed as

$$\mathbf{L}^{(t)} = \mathbf{M} \bar{\mathbf{P}}_s^{(t)} \mathbf{M}^T + \mathbf{Z}_{\text{noise}}, \quad (21)$$

where $\mathbf{Z}_{\text{noise}}$ is the measurement of noise covariance matrix. Finally, the state covariance at t -th time can be updated by

$$\mathbf{P}_s^{(t)} = (\mathbf{I} - K^{(t)} \mathbf{M}) \bar{\mathbf{P}}_s^{(t)}, \quad (22)$$

where $\mathbf{I}$ denotes an identity matrix.

3.2 UAV Navigation

Let us assume the location of the UAV, which is denoted by g in the $\mathbf{W}$ environment, is estimated using the aforementioned Kalman filter algorithm. The UAV is located at $(x, y, z) \in \mathbf{W}$ at time t , and can be expressed as

$$g_t = (x_t, y_t, z_t), \forall t \in [0, T]. \quad (23)$$

At the initial time $t=0$, the location of the UAV in an unknown environment $\mathbf{W}$ is initialized at $(0, 0, 0)$ which can be expressed as

$$g_0 = (0, 0, 0). \quad (24)$$

In the navigation process, the UAV moves from the initial point X_0 to the goal point X_r . The displacement of the UAV generates a trajectory. The trajectory of the UAV in the map of environment $\mathbf{W}$ can be expressed as

$$G_{0:T} : \{g_0, g_1, g_2, \dots, g_T\}. \quad (25)$$

Meanwhile, the motion of the UAV r_t during the flight maneuver at time t is given by

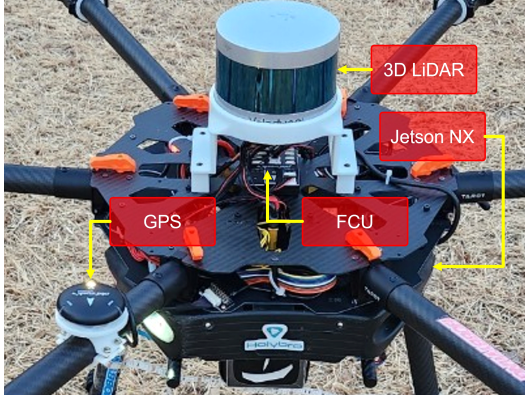


Fig. 4. Hardware implementation of UAV 3D LiDAR avoidance.

$$r_{0:T} : \{r_0, r_1, r_2, \dots, r_T\}. \quad (26)$$

As the environment condition $\mathbf{W}$ and location g_t of the UAV are updated over time, the environment W is observed from the initial time t_0 to time T . It can be expressed as

$$o_{0:T} : \{o_0, o_1, o_2, \dots, o_T\}. \quad (27)$$

Subsequently, the trajectory of the UAV and map are estimated by

$$p(G_{0:T}, \mathbf{W} | o_{1:T}, r_{1:T}), \quad (28)$$

where p denotes the probability distribution of the trajectory of the UAV $G_b \forall t \in [0, T]$ and the environment $\mathbf{W}$ based on the motion control (o_1, r_1) at the next time step $t+1$.

The UAV moves from the initial point X_0 to a goal point X_f through an obstacle in environment $\mathbf{W}$. The path planning for navigation is generated based on the estimated map environment $\mathbf{W}$ and location g_t of the UAV.

To generate a path, the environmental condition $\mathbf{W} \subset \mathbb{R}^3$ should acknowledge the free and occupied space in the map. A point in the 3D space $f_k(x_b, y_b, z_b) \in \mathbb{R}^3$ is defined as occupied if the point is filled with a point cloud $\mathbf{P}_k(x_b, y_b, z_b) \in C$. Otherwise, it is defined as an accessible point. This can be expressed as

Table 3. Avoidance success-rate experimental setup.

Parameter	Value
Weather	Sunny, Outdoor
Temperature (° C)	28 - 32
Wind speed (m/s)	2 - 3
Trial (count)	10
Distance (count)	9
Observation time each distance (s)	10
Observation time each trial (s)	90
Distance GCS - UAV (m)	5
UAV altitude (m)	3.5

$$f_k(x_k, y_k, z_k) = \begin{cases} 1, & \text{if } f_k = \mathbf{P}_k \\ 0, & \text{otherwise.} \end{cases}$$

A set of occupied and free points $f_b \forall k \in K$ is called an occupied and accessible space, respectively. An occupied space implies that an obstacle is located in that area, whereas a free space indicates that the area is free from obstacles. The occupied and free spaces are presented as occupancy grids.

IV. Experimental Results and Analysis

The performance of the proposed algorithm is validated through experiments conducted in real environments. The experimental setup can be seen in Fig. 3, where an Nvidia Jetson NX was utilized as an onboard PC to process the point cloud data from the Velodyne 3D LiDAR in real time. In order to minimize delay during data transmission, the onboard PC and the 3D LiDAR were connected using ethernet. The 3D LiDAR point cloud data processing and real-time generated UAV trajectory on the onboard PC were monitored using a laptop on the ground. This study employs a WiFi router connection to maintain a stable connection between ROS in the onboard PC and the ground control station. Fig. 4 depicts a configuration of sensors, onboard PC, and the UAV setup. A hexacopter tarot X6 was employed to enable carrying the loads. The 3D LiDAR, GPS sensor, onboard PC, and Pixhawk flight controller unit were loaded on the UAV. The proposed algorithm was im-

plemented in Python. To be specified, the whole proposed algorithm was built and implemented in the Ardupilot and robot operating system (ROS) Melodic environment.

Figure 5(a) depicts an average detection time in milliseconds (ms) obtained from real time 3D LiDAR mapping from [31], current proposed system, and existing 2D LiDAR avoidance in [32]. The experiment was conducted by deploying an obstacle at random different angle of 8 sector and 20 meters distance with 10 times repetition. Based on obtained result, the proposed system is outperformed^[31] in term of average detection rate, since the proposed sector filtering algorithm reduces the total number of PCL. However, the trade-off of using the 3D LiDAR caused a higher detection time compared with existing 2D LiDAR avoidance in [32]. This is due to the lower resolution of 2D LiDAR scan; thus, the existing 2D based avoidance requires a lower computational complexity.

Figure 5(b) presents the obstacle detection employing a sector-filtering algorithm and the results are compared with the real distance measurement. The experimental setup is described as follows:

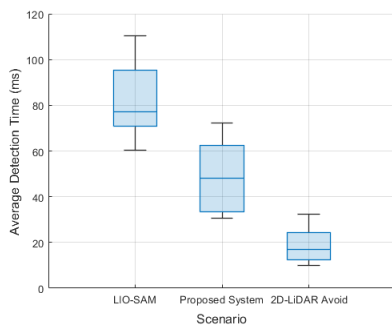
- Consider low altitude UAV to calculate the error between measured and real distance;
- The experiment conducted by moving an obstacle at a fixed distance from one sector to another sector generated by proposed algorithm, therefore, the time elapsed for one distance and varying sector is 10 seconds;
- To validate the observation, this study consider

10 trials (random 8 sectors selection) for each fixed distance and approximate measurement by taking mean value;

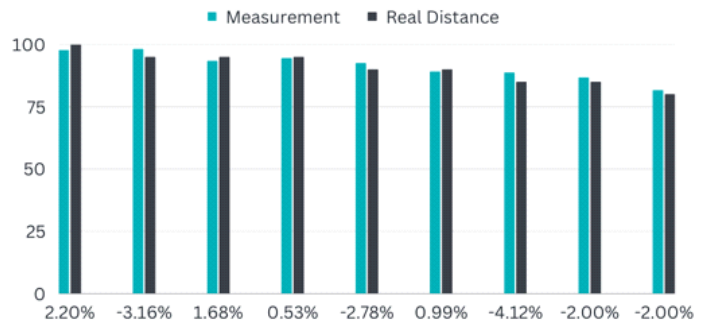
- This experiment considers 9 variable distances which are [100, 98, 97, 96, 92, 90, 82, 80, 77] meters away from LiDAR.

As shown in the figure, the maximum error of obstacle detection employing the sector-filtering algorithm is below 4% in all the cases. The results show that the proposed sector-filtering algorithm could detect obstacles correctly in real-time.

Fig. 6(a) depicts the real-world experiment of filtered 3D LiDAR point cloud. It can be seen that the red dot indicates a laser scan ROS message that is obtained from the proposed algorithm. In contrast with Fig. 1, the obtained laser scan is less populated with a 3D point cloud. Therefore, it can reduce the computational burden of UAV while estimating the path that has to be taken to reach the destination. Fig. 6(b) depicts an experiment visualization of obstacle detection through the proposed algorithm. It can be seen that the algorithm will filter the 3D point cloud and divide it into 8 sectors. The red curve indicates the drone's distance from the closest point cloud, and every sector will move dynamically according to the filtered point cloud. Moreover, for ease of use, the obtained distance is measured in meters and the sector angle in degrees. Fig. 6(c) shows that the maximum distance of a detected obstacle can go up to 100 meters. Therefore, UAVs can efficiently estimate and calculate the path from start to destination. In addition, with the re-



(a)



(b)

Fig. 5. (a) Average detection time obtained from 10 trials for different scheme. (b) Average accuracy obtained from 20 trials for each trial distance.

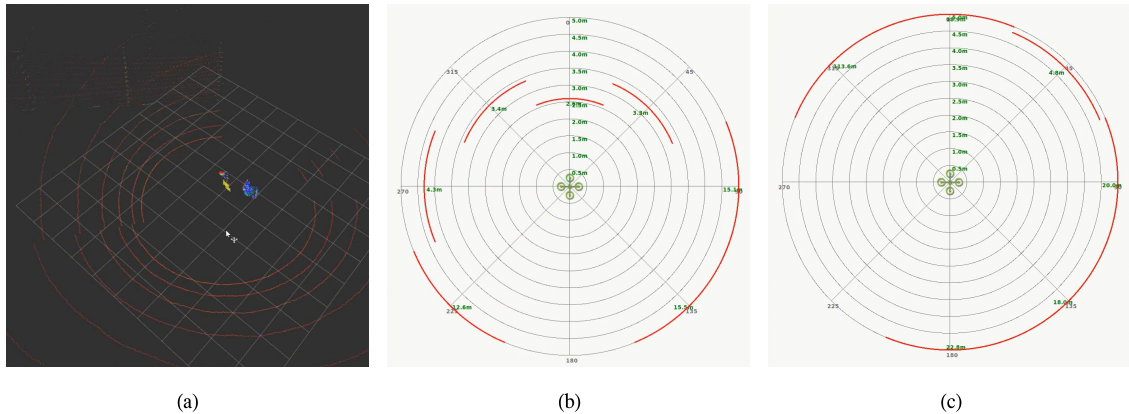


Fig. 6. (a) Detected object/obstacle from 8 sectors of point cloud. (b) RVIZ visualization of 360° of 2D laser scan ROS message. (c) The detection range of the proposed system can work up to 100 meters.

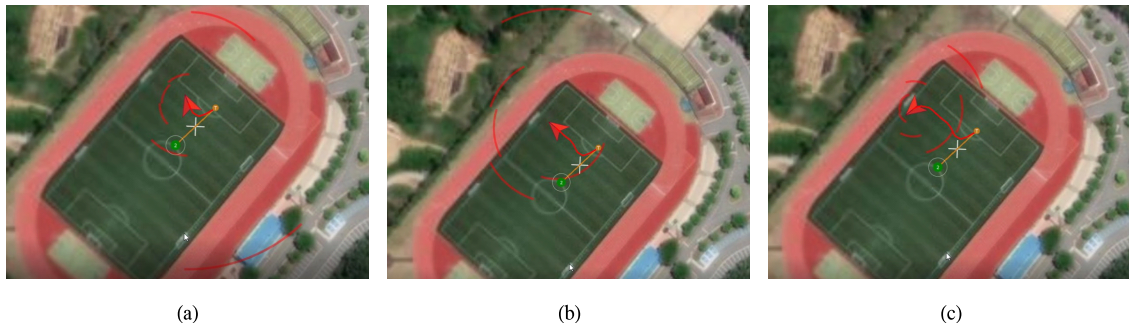


Fig. 7. The real-world implementation of the proposed system. The experimental results of UAV trajectory which (a) avoiding an obstacle, (b) finding a new path, (c) and continuing path to the end waypoint.

al-time 8-sector estimation, the UAV can easily evade an incoming obstacle due to its long-range detection capability.

Fig. 7 shows the real-world implementation of the proposed system in the hexacopter UAV. As can be seen from the figure, the UAV detects an obstacle after flying from the initial location, successfully avoids the obstacle, and then continues heading toward the target location. For ease of understanding, Fig. 7(a) shows the UAV successfully detects an obstacle after flying from the initial location marked as an orange dot. In addition, a red curve that appears in this follows Fig. 6(b) behavior, where each red curve represents generated sector filtering and distance from UAV to obstacle. Fig. 7(b) shows the UAV changed its flight directions to avoid the detected obstacle marked by a red triangle and red path as the trajectory of the UAV. Moreover, as shown in the figure, the UAV continuously senses obstacles around its position. Furthermore, Fig. 7(c) shows the UAV

flies toward the target location marked as a green dot after successfully avoiding the obstacle while sensing the other obstacles.

V. Conclusion

This paper employed a sector-filtering algorithm to improve real-time obstacle avoidance of UAV using 3D LiDAR. The sector-filtering algorithm mitigated the high-computational complexity issue of processing high-resolution and massive point cloud data from 3D LiDAR at a time by employing sectoring and filtering processes. Therefore, the computational time for dynamic obstacle detection in the 360-degree FoV was significantly reduced, and real-time obstacle avoidance was performed. The proposed algorithm was validated through experiments conducted in a real environment. The results show that the obstacles have been successfully detected within the 100-meter range, and the UAV avoidance maneuver is performed

in real time. The future research directions are described as follows:

- Due to the limitation of the current system that there is no mapping process, it will be difficult for the UAV to maneuver in complex environments. The integration of mapping and optimized pathplanning is considered as an appealing future research direction.
- A larger number of UAVs with a cooperative scheme such as multi-cooperative UAVs has potentially integrated with the proposed sector-filtering algorithm in each UAV.

References

- [1] M. A. Tahir, I. Mir, and T. U. Islam, "A review of UAV platforms for autonomous applications: Comprehensive analysis and future directions," *IEEE Access*, vol. 11, pp. 52540-52554, May 2023.
- [2] S. Rezwan and W. Choi, "Artificial intelligence approaches for UAV navigation: Recent advances and future challenges," *IEEE Access*, vol. 10, pp. 26320-26339, Mar. 2022.
- [3] N. Nomikos, P. K. Gkonis, P. S. Bithas, and P. Trakadas, "A survey on UAV-aided maritime communications: Deployment considerations, applications, and future challenges," *IEEE Open J. Commun. Soc.*, vol. 4, pp. 56-78, Jan. 2023.
- [4] P. McEnroe, S. Wang, and M. Liyanage, "A survey on the convergence of edge computing and AI for UAVs: Opportunities and challenges," *IEEE Internet Things J.*, vol. 9, no. 17, pp. 15435-15459, Sep. 2022.
- [5] X. Wang, L. T. Yang, D. Meng, M. Dong, K. Ota, and H. Wang, "Multi-UAV cooperative localization for marine targets based on weighted subspace fitting in SAGIN environment," *IEEE Internet Things J.*, vol. 9, no. 8, pp. 5708-5718, Apr. 2022.
- [6] Y. Xue and W. Chen, "A UAV navigation approach based on deep reinforcement learning in large cluttered 3D environments," *IEEE Trans. Veh. Technol.*, vol. 72, no. 3, pp. 3001-3014, Mar. 2023.
- [7] C. Mei, Z. Fan, Q. Zhu, P. Yang, Z. Hou, and H. Jin, "A novel scene matching navigation system for UAVs based on vision/inertial fusion," *IEEE Sensors J.*, vol. 23, no. 6, pp. 6192-6203, Mar. 2023.
- [8] J. Dong, X. Ren, S. Han, and S. Luo, "UAV vision aided INS/odometer integration for land vehicle autonomous navigation," *IEEE Trans. Veh. Technol.*, vol. 71, no. 5, pp. 4825-4840, May 2022.
- [9] C. Yang, Z. Xiong, J. Liu, L. Chao, and Y. Chen, "A path integration approach based on multiscale grid cells for large-scale navigation," *IEEE Trans. Cogn. Develop. Sys.*, vol. 14, no. 3, pp. 1009-1020, Sep. 2022.
- [10] H. Dong, C.-Y. Weng, C. Guo, H. Yu, and I.-M. Chen, "Real-time avoidance strategy of dynamic obstacles via half model-free detection and tracking with 2D lidar for mobile robots," *IEEE/ASME Trans. Mechatronics*, vol. 26, no. 4, pp. 2215-2225, Aug. 2021.
- [11] B. Lindqvist, S. S. Mansouri, J. Haluška, and G. Nikolakopoulos, "Reactive navigation of an unmanned aerial vehicle with perception-based obstacle avoidance constraints," *IEEE Trans. Control Syst. Technol.*, vol. 30, no. 5, pp. 1847-1862, Sep. 2022.
- [12] J. Liang, Y.-L. Qiao, T. Guan, and D. Manocha, "OF-VO: Efficient navigation among pedestrians using commodity sensors," *IEEE Robotics Automat. Lett.*, vol. 6, no. 4, pp. 6148-6155, Oct. 2021.
- [13] A. Torii, et al., "Are large-scale 3D models really necessary for accurate visual localization?," *IEEE Trans. Pattern Anal. Machine Intel.*, vol. 43, no. 3, pp. 814-829, Mar. 2021.
- [14] Z. Zhou, et al., "SGM3D: Stereo guided monocular 3D object detection," *IEEE Robotics Automat. Lett.*, vol. 7, no. 4, pp. 10478-10485, Oct. 2022.
- [15] D. Falanga, S. Kim, and D. Scaramuzza, "How fast is too fast? the role of perception

- latency in high-speed sense and avoid,” *IEEE Robotics Automat. Lett.*, vol. 4, no. 2, pp. 1884-1891, Apr. 2019.
- [16] G. Chen, W. Dong, X. Sheng, X. Zhu, and H. Ding, “An active sense and avoid system for flying robots in dynamic environments,” *IEEE/ ASME Trans. Mechatronics*, vol. 26, no. 2, pp. 668-678, Apr. 2021.
- [17] V. Tadić, et al., “Application of the ZED depth sensor for painting robot vision system development,” *IEEE Access*, vol. 9, pp. 117845-117859, Aug. 2021.
- [18] J. Muhovič, R. Mandeljc, B. Bovcon, M. Kristan, and J. Perš, “Obstacle tracking for unmanned surface vessels using 3-D point cloud,” *IEEE J. Oceanic Eng.*, vol. 45, no. 3, pp. 786-798, Jul. 2020.
- [19] D. Dodge and M. Yilmaz, “Convex vision-based negative obstacle detection framework for autonomous vehicles,” *IEEE Trans. Intel. Veh.*, vol. 8, no. 1, pp. 778-789, Jan. 2023.
- [20] C. Wisultschew, G. Mujica, J. M. Lanza Gutierrez, and J. Portilla, “3D-LIDAR based object detection and tracking on the edge of IoT for railway level crossing,” *IEEE Access*, vol. 9, pp. 35718-35729, Mar. 2021.
- [21] J. Lin, H. Yin, J. Yan, W. Ge, H. Zhang, and G. Rigoll, “Improved 3D object detector under snowfall weather condition based on LiDAR point cloud,” *IEEE Sensors J.*, vol. 22, no. 16, pp. 16276-16292, Aug. 2022.
- [22] Y. Sun, W. Zuo, H. Huang, P. Cai, and M. Liu, “PointMoSeg: Sparse tensor-based end-to-end moving-obstacle segmentation in 3D lidar point clouds for autonomous driving,” *IEEE Robotics Automat. Lett.*, vol. 6, no. 2, pp. 510-517, Apr. 2021.
- [23] F. Kong, W. Xu, Y. Cai, and F. Zhang, “Avoiding dynamic small obstacles with on-board sensing and computation on aerial robots,” *IEEE Robotics Automat. Lett.*, vol. 6, no. 4, pp. 7869-7876, Oct. 2021.
- [24] J. Fang, et al., “Augmented LiDAR simulator for autonomous driving,” *IEEE Robotics Automat. Lett.*, vol. 5, no. 2, pp. 1931-1938, Apr. 2020.
- [25] C. Wen, A. F. Habib, J. Li, C. K. Toth, C. Wang, and H. Fan, “Special issue on 3D sensing in intelligent transportation,” *IEEE Trans. Intel. Transport. Sys.*, vol. 22, no. 4, pp. 1947-1949, Apr. 2021.
- [26] Z. Rozsa and T. Sziranyi, “Obstacle prediction for automated guided vehicles based on point clouds measured by a tilted LIDAR sensor,” *IEEE Trans. Intel. Transport. Sys.*, vol. 19, no. 8, pp. 2708-2720, Aug. 2018.
- [27] B. Wang, M. Zhu, Y. Lu, J. Wang, W. Gao, and H. Wei, “Real-time 3D object detection from point cloud through foreground segmentation,” *IEEE Access*, vol. 9, pp. 84886-84898, Jun. 2021.
- [28] Open Source Robotics Foundation, *Robotic Operating System Point Cloud Message*. http://wiki.ros.org/pcl_ros
- [29] Open Source Robotics Foundation, *Velodyne Robotic Operating System Documentation*, <https://wiki.ros.org/velodyne>
- [30] Y. Jin, et. al., “Filtering processing of LiDAR point cloud data,” *IOP Conf. Ser.: Earth Environ. Sci.*, 783,012125, 2021.
- [31] ArduPilot Dev Team, “Path Planning and Obstacle Avoidance Features,” <https://ardupilot.org/copter/docs/common-object-avoidance-landing-page.html>, access on: 5 Jul. 2024.
- [32] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, “LIO-SAM: Tightly-coupled lidar inertial odometry via smoothing and mapping,” *2020 IEEE/RSJ Int. Conf. IROS*, pp. 5135-5142, Las Vegas, NV, USA, 2020. (<https://doi.org/10.1109/IROS45743.2020.9341176>)

Krisma Asmoro



Sept. 2018 : B.S. degree, Telkom University

Sept. 2020~current : Integrated Master and PhD. student, Kumoh National Institute of Technology

<Research Interests> 5G / B6G

communication, multiple access technology, computer vision, autonomous vehicle communication

[ORCID:0000-0001-6053-2736]

Ryan Febriansyah



Feb. 2020 : B.S.degree, Telkom University

Feb. 2023 : M.S.degree, Kumoh National Institute of Technology

2023~current : Research and development at Rovostech Co., Ltd, South Korea

<Research Interests> Robotics, unmanned vehicle, computer vision and pattern recognition

[ORCID:0009-0004-4216-4211]

Silvirianti



Feb. 2017 : B.S.degree, Telkom University

Feb. 2020 : M.S.degree, Kumoh National Institute of Technology

Sept. 2023 : Ph.D. degree, Kumoh National Institute of Technology

2024~current : Post Doctoral researcher at École de Technologie Supérieure, Canada

<Research Interests> include UAV communications, nonorthogonal multiple access, energy efficient communication, robotics, and autonomous vehicle

[ORCID:0000-0001-6933-2802]

Soo Young Shin



Feb. 1999 : B.S.degree, Seoul National University

Feb. 2001 : M.S.degree, Seoul National University

Feb. 2006 : Ph.D., Seoul National University

Sept. 2010~current : Professor, Kumoh National Institute of Technology

<Research Interests> 5G/6G wireless communications, Internet of things, drone applications

[ORCID:0000-0002-2526-239]